This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Currently amended) A method for operating a computer using object-based computer code of an object-oriented programming language, the method comprising:

utilizing an explicit interface member mechanism that enables at least one software component to implement at least one explicit interface member by explicitly specifying the relationship between said at least one software component and the at least one explicit interface member[[.]]; and

storing said at least one software component in a form that includes said implemented at least one explicit interface member in a computer readable storage medium.

2. (Original) A method according to claim 1, wherein said specifying of the relationship includes specifying a qualified name of the at least one software component.

3. (Original) A method according to claim 2, wherein said specifying of the qualified name includes specifying at least one interface name and said at least one interface member name.

4. (Original) A method according to claim 1, wherein said explicit interface member mechanism enables an explicit interface member implementation to be excluded from the public interface of said at least one software component.

5. (Original) A method according to claim 1, wherein the at least one software component is at least one of a class or struct instance, as defined by the object-oriented programming language.

6. (Original) A method according to claim 1, wherein the explicit interface member mechanism enables said at least one software component to implement an internal interface not accessible to a consumer of said at least one software component.

7. (Original) A method according to claim 1, wherein said explicit interface member mechanism enables disambiguation of a plurality of interface members having the same signature.

8. (Original) A method according to claim 1, wherein said explicit member mechanism enables disambiguation of a plurality of interface members having the same signature and return type.

9. (Original) A method according to claim 1, wherein in addition to allowing the implementation of public interface members, said explicit interface member mechanism enables the implementation of private interface members.

10. (Original) A method according to claim 1, wherein said explicit interface member mechanism enables the implementation of a plurality of non-conflicting specific versions of a generic interface.

11. (Original) A method according to claim 1, wherein the computer code is programmed according to an object-oriented programming language, and said object-oriented programming language is one of C#, Fortran, Pascal, Visual Basic, C, C++ and Java.

12. (Original) A method according to claim 1, wherein an implementation of an explicit interface member is a method, property, event, or indexer declaration that references a fully qualified interface member name.

13. (Original) A method according to claim 1, wherein said at least one software component names an interface in [[the]] a base class list of the at least one software component that contains a member whose fully qualified name, type, and parameter types exactly match those of the implementation of the explicit interface member.

14. (Original) A method according to claim 1, wherein said explicit interface member mechanism includes an interface mapping mechanism that locates implementations of interface members in said at least one software component.

15. (Currently amended) A method according to claim 14, wherein said interface mapping mechanism locates an implementation for each member of each interface specified in [[the]] a base class list of the at least one software component.

16. (Currently amended) A method according to claim 1, wherein said at least one software component inherits all interface implementations provided by its base classes.

17. (Original) A method according to claim 1, wherein it is not possible to override an explicit interface member implementation, but where an explicit interface member implementation calls another virtual method, derived classes are capable of overriding the implementation.

18. (Original) A method according to claim 1, wherein a software component of said at least one software component that inherits an interface implementation is permitted to re-implement the interface by including the interface in the base class list of the software component.

19. (Original) A method according to claim 1, wherein said explicit interface member mechanism prevents conflict among specific implementations of a generic interface.

20. (Currently amended) A computer readable storage medium bearing computer executable instructions for carrying out the method of claim 1.

21. (Canceled)

22. (Original) A computing device comprising means for performing the method of claim 1.

23. (Currently amended) A computer readable storage medium having stored thereon a plurality of computer-executable modules written in an object-oriented programming language, the computer executable modules comprising:

an explicit interface member mechanism that enables at least one software component to implement at least one explicit interface member by explicitly specifying the relationship between said at least one software component and the at least one interface member.

24. (Currently amended) A computer readable storage medium according to claim 23, wherein said specifying of the relationship includes specifying a qualified name of the at least one software component.

25. (Currently amended) A computer readable storage medium according to claim 24, wherein said specifying of the qualified name includes specifying at least one interface name and said at least one interface member name.

26. (Currently amended) A computer readable storage medium according to claim 23, wherein said explicit interface member mechanism enables an explicit interface member implementation to be excluded from the public interface of said at least one software component.

27. (Currently amended) A computer readable storage medium according to claim 23, wherein the at least one software component is at least one of a class or struct instance, as defined by the object-oriented programming language.

28. (Currently amended) A computer readable storage medium according to claim 23, wherein the explicit interface member mechanism enables said at least one software component to implement an internal interface not accessible to a consumer of said at least one software component.

29. (Currently amended) A computer readable storage medium according to claim 23, wherein said explicit interface member mechanism enables disambiguation of a plurality of interface members having the same signature.

30. (Currently amended) A computer readable storage medium according to claim 23, wherein said explicit member mechanism enables disambiguation of a plurality of interface members having the same signature and return type.

31. (Currently amended) A computer readable storage medium according to claim 23, wherein in addition to allowing the implementation of public interface members, said explicit interface member mechanism enables the implementation of private interface members.

32. (Currently amended) A computer readable storage medium according to claim 23, wherein said explicit interface member mechanism enables the implementation of a plurality of non-conflicting specific versions of a generic interface.

33. (Currently amended) A computer readable storage medium according to claim 23, wherein the object-oriented programming language is one of C#, Fortran, Pascal, Visual Basic, C, C++ and Java.

34. (Currently amended) A computer readable storage medium according to claim 23, wherein an implementation of an explicit interface member is a method, property, event, or indexer declaration that references a fully qualified interface member name.

35. (Currently amended) A computer readable storage medium according to claim 23, wherein said at least one software component names an interface in the base class list of the at least one software component that contains a member whose fully qualified name, type, and parameter types exactly match those of the implementation of the explicit interface member.

36. (Currently amended) A computer readable storage medium according to claim 23, wherein said explicit interface member mechanism includes an interface mapping mechanism that locates implementations of interface members in said at least one software component.

37. (Currently amended) A computer readable storage medium according to claim 36, wherein said interface mapping mechanism locates an implementation for each member of each interface specified in [[the]] a base class list of the at least one software component.

38. (Currently amended) A computer readable storage medium according to claim 23, wherein said at least one software component inherits all interface implementations provided by its base classes.

39. (Currently amended) A computer readable storage medium according to claim 23, wherein it is not possible to override an explicit interface member implementation, but where an explicit interface member implementation calls another virtual method, derived classes are enabled to override the implementation.

40. (Currently amended) A computer readable storage medium according to claim 23, wherein a software component of said at least one software component that inherits an interface implementation is permitted to re-implement the interface by including it in [[the]] a base class list of the software component.

41. (Currently amended) A computer readable storage medium according to claim 23, wherein said explicit interface member mechanism prevents conflict among specific implementations of a generic interface.

42. (Canceled)

43. (Canceled)

44. (Canceled)

45. (Canceled)

46. (Canceled)

47. (Canceled)

48. (Canceled)

49. (Canceled)

50. (Canceled)

51. (Canceled)

52. (Canceled)

53. (Canceled)

54. (Canceled)

55. (Canceled)

56. (Canceled)

57. (Canceled)

58. (Currently amended) ~~An object-oriented programming language~~ The method ~~according to~~ of claim [[42]] 61, wherein it is not possible to override an explicit interface member implementation, but where an explicit interface member implementation calls another virtual method, derived classes are enabled to override the implementation.

59. (Currently amended) ~~An object-oriented programming language~~ The method ~~according to~~ of claim [[42]] 61, wherein a software component of said ~~at least one~~ software component that inherits an interface implementation is permitted to re-implement the interface by including it in the base class list of the software component.

60. (Currently amended) ~~An object-oriented programming language~~ The method ~~according to~~ of claim [[42]] 61, wherein said implementing said explicit interface member ~~mechanism~~ prevents conflict among specific implementations of a generic interface.

61. (New) A method of generating an object comprising:

implementing an explicit interface member in a software component by specifying a relationship between said software component and an interface member; and

initializing an instance of said software component in a computer readable storage medium.

62. (New) The method of claim 61, wherein said specifying a relationship includes specifying a qualified name of the software component.

63. (New) The method of claim 62, wherein said specifying a qualified name includes specifying an interface name and said interface member name.

64. (New) The method of claim 61 wherein said implemented explicit interface member is excluded from the public interface of said software component.

65. (New) The method of claim 61, wherein the software component is at least one of a class or struct.

66. (New) The method of claim 61, wherein said software component implements an internal interface that is not accessible to a consumer of said software component.

67. (New) The method of claim 61 wherein said implemented explicit interface member enables disambiguation of interface members with the same signature.

68. (New) The method of claim 61 wherein said implemented explicit interface member enables disambiguation of interface members with the same signature and return type.

69. (New) The method of claim 61 wherein said implemented explicit interface member enables the implementation of private interface members.

70. (New) The method of claim 61, wherein said implemented explicit interface member enables a plurality of non-conflicting specific versions of a generic interface.

71. (New) The method of claim 61, wherein the explicit interface member is implemented in an object-oriented programming language.

72. (New) The method of claim 61, wherein said implemented explicit interface member is a member selected from a group of members consisting of a method, a property, an event, and an indexer declaration that references a fully qualified interface member name.

73. (New) The method of claim 61, wherein said software component names an interface in a base class list of the software component that contains a member whose fully qualified name, type, and parameter types exactly match those of the implemented explicit interface member.

74. (New) The method of claim 54 further comprising:

locating implemented interface members in each interface in said software component.

75. (New) The method of claim 66, wherein said locating implementations of each interface member includes locating an implementation for each member of each interface specified in a base class list of the software component.

76. (New) The method of claim 61, wherein said software component inherits all interface implementations provided by its base classes.